

Claims

What is claimed is :

1. a method for implementing autonomous load/store within a data processing system by using symbolic machine code,

where said data processing system comprises a microprocessor and a memory system,

where said memory system has a memory hierarchy containing :

- one or more register files of said microprocessor
- one or more data caches at different memory hierarchy levels
- a main memory

where said microprocessor has an instruction set and where said instruction set contains one or more instructions of which one or more operands and/or results may specify one or more symbolic variables,

where a symbolic machine code is running on said microprocessor,

where said symbolic machine code contains one or more instructions of which one or more operands and/or results specify one or more symbolic variables,

where said autonomous load/store refers to the loading and storing of data generated and used by said symbolic machine code and where at least part of said data loading and storing is done without requiring any explicit load/store instructions in said symbolic machine code,

where each of said symbolic variables specifies one or more entries of a memory other than a register file of said microprocessor,

where said entries are used by the microprocessor in order to determine the addresses within the memory hierarchy where the values of said symbolic variables may be stored to and/or loaded from during execution of said symbolic machine code,

where said method comprises the following steps :

- a. when the microprocessor fetches an instruction of which a result and/or one or more operands specify one or more symbolic variables, it computes the addresses within the memory hierarchy where the values of said symbolic variables may be stored into and/or loaded from;
- b. after anyone of said addresses has been computed, the microprocessor writes this computed address into an entry of a dedicated memory; in the following, said dedicated memory is referred to by the term heap address cache;
- c. in addition to said computed address of step b., said microprocessor writes data associated to said computed address into said heap address cache and/or into another memory; said data are also called link data in the following; said link data are such that, when they are accessed by the microprocessor, they allow the microprocessor to make the link with or to associate them to said computed address and may be used to determine whether said computed address refers to the value of an operand and/or of a result of said instruction
- d. the microprocessor uses said entry within said heap address cache in order to determine or estimate :

- i. the lifetime of the value to be stored to or to be loaded from said computed address
 - ii. and/or the amount of time which elapsed since a previous write of the same address into an entry of said heap address cache;
2. a method as claimed in claim 1,
where step d. is further specified as follows :
the microprocessor uses said entry within said heap address cache and/or said link data in order to determine or estimate the lifetime of said value by determining or estimating the amount of time which elapsed since a previous write of the same address into an entry of said heap address cache;
3. a method as claimed in claim 2,
where said heap address cache is realized as a circular stack,
where steps b. and c. are further specified as follows :
 - b. after anyone of said addresses has been computed, the microprocessor writes this computed address into the same entry of the circular stack as the one where the link data in step d. are written;
 - c. said microprocessor writes said link data into the same entry of the circular stack as said computed address, said link data comprising one or both of the following :
 - a type-flag, which tells the microprocessor whether the value stored or to be stored at said computed address refers to the value of an instruction operand or of an instruction result
 - a valid-flag, which tells the microprocessor whether the entry contains data which can be overwritten or not
4. a method as claimed in claim 3,
where, in addition to the data mentioned in step c., said link data further contain an execution state value, this value allowing to determine the execution state of said symbolic machine code at the point in time when said instruction is fetched or when said link data are written
5. a method as claimed in claim 3,
where step b. is further specified as follows :
the microprocessor uses said entry within said heap address cache and/or said link data in order to determine or estimate the lifetime of said value by subtracting the entry containing said computed address from another entry of said heap address cache containing the same address;
6. a method as claimed in claim 4,
where step b. is further specified as follows :
the microprocessor uses said entry within said heap address cache and said execution state value in order to determine or estimate the lifetime of said value by subtracting the execution state value stored in the entry containing said computed address from the execution state value stored in the same or in another entry of said heap address cache containing the same address;

7. a method as claimed in claim 1,
where the microprocessor uses the lifetimes of the values of said symbolic variables in order to determine the addresses and/or the hierarchy levels within the memory hierarchy where said values shall be stored;
8. a method as claimed in claim 2,
where the microprocessor uses the lifetimes of the values of said symbolic variables in order to determine the addresses and/or the hierarchy levels within the memory hierarchy where said values shall be stored;
9. a method as claimed in claim 3,
where the microprocessor uses the lifetimes of the values of said symbolic variables in order to determine the addresses and/or the hierarchy levels within the memory hierarchy where said values shall be stored;
10. a method as claimed in claim 4,
where the microprocessor uses the lifetimes of the values of said symbolic variables in order to determine the addresses and/or the hierarchy levels within the memory hierarchy where said values shall be stored;
11. a method as claimed in claim 5,
where the microprocessor uses the lifetimes of the values of said symbolic variables in order to determine the addresses and/or the hierarchy levels within the memory hierarchy where said values shall be stored;
12. a method as claimed in claim 6,
where the microprocessor uses the lifetimes of the values of said symbolic variables in order to determine the addresses and/or the hierarchy levels within the memory hierarchy where said values shall be stored;
13. A microprocessor having an instruction set containing :
 - one or more instructions of which one or more operands and/or results may specify one or more symbolic variables
 - one or more symbolic link instructionswhere said microprocessor is able to execute symbolic machine code,
where said symbolic machine code contains one or more instructions of which one or more operands and/or results specify one or more symbolic variables